CMPG-767 Digital Image Processing

SPATIAL DOMAIN FILTERING
EDGE DETECTION

Edges

- Edge pixels are pixels at which the intensity changes abruptly
- Edges are sets of connected edge pixels
- Often edges are called points of discontinuity of the intensity function
- To detect edges, differentiation in a local neighborhood should be used

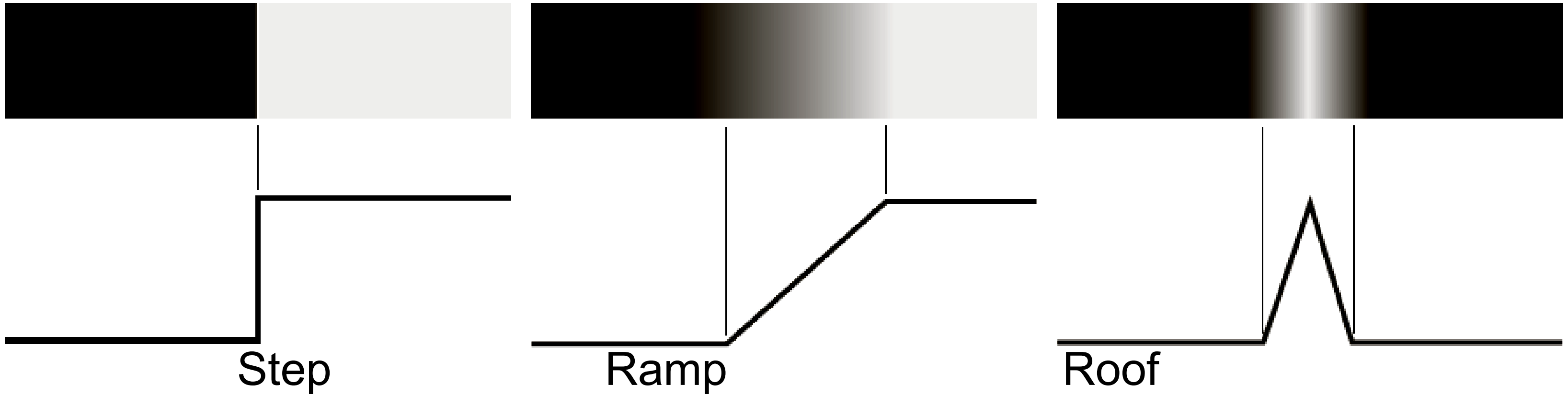
Importance of Edge Detection

- Edge detection is important for localization of image details, especially the smallest details
- It can be used, for example, in non- destructive inspection, to localize possible defects; in medical imaging, to localize important details
- Edge detection is also used in image segmentation – a process that partitions an image into disjoint regions corresponding to different textures

High-Pass Filtering

- In terms of filtering, edge detection is a spatial domain high-pass filtering
- Differentiating an image, high-pass filters eliminate low and medium frequencies, passing and possibly enhancing high frequencies.

Three Major Types of Intensity Change



a b c

FIGURE 10.8
From left to right, models (ideal representations) of a step, a ramp, and a roof edge, and their corresponding intensity profiles.

Derivatives of Digital Functions

- The derivatives of a digital image function are defined in terms of differences. They have to meet the following requirements:
- A 1st derivative

1) Must be 0 in areas of constant intensity
2) Must be nonzero at the onset of an intensity step or ramp;
3) Must be nonzero along ramps

Derivatives of Digital Functions

- The derivatives of a digital image function are defined in terms of differences. They have to meet the following requirements:
- A 2nd derivative

1) Must be 0 in areas of constant intensity
2) Must be nonzero at the onset and end of an intensity step or ramp;
3) Must be zero along ramps of constant slope

Derivatives of 1-D Digital Functions

- A basic definition of the first-order derivative of the digital function $f(x), x = 0, 1, \ldots, N-1$

$$\frac{\partial f}{\partial x} = \frac{\Delta f}{\Delta x} = \frac{f(x+1) - f(x)}{x+1-x} = \frac{f(x+1) - f(x)}{1} = f(x+1) - f(x)$$

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

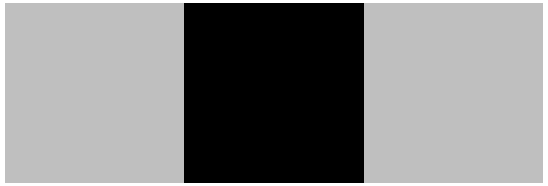
Derivatives of 1-D Digital Functions

- A basic definition of the second-order derivative of the digital function $f(x), x = 0, 1, \ldots, N-1$

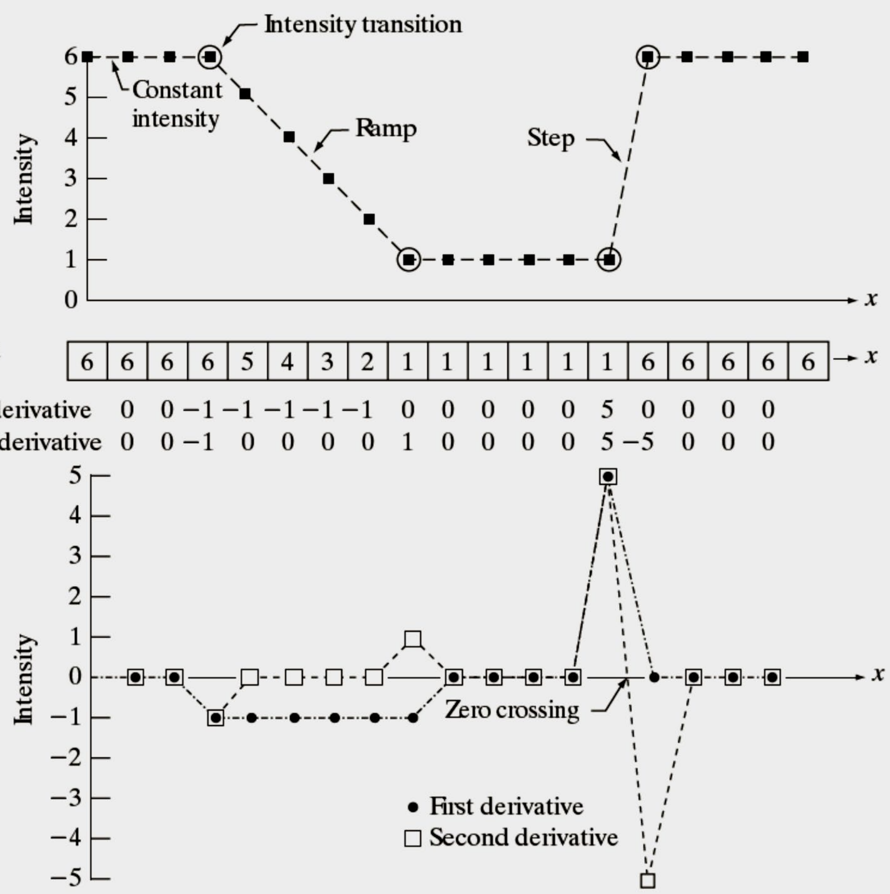
$$\frac{\partial^2 f}{\partial x^2} = \left(f(x+1) - f(x)\right) - \left(f(x) - f(x-1)\right) =$$

$$= f(x+1) + f(x-1) - 2f(x)$$

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x)$$



Derivatives of 1-D Digital Functions



Scan line	6	6	6	6	5	4	3	2	1	1	1	1	1	1	6	6	6	6	6
1st derivative		0	0	−1	−1	−1	−1	−1	0	0	0	0	0	5	0	0	0	0	
2nd derivative		0	0	−1	0	0	0	0	1	0	0	0	0	5	−5	0	0	0	

a
b
c

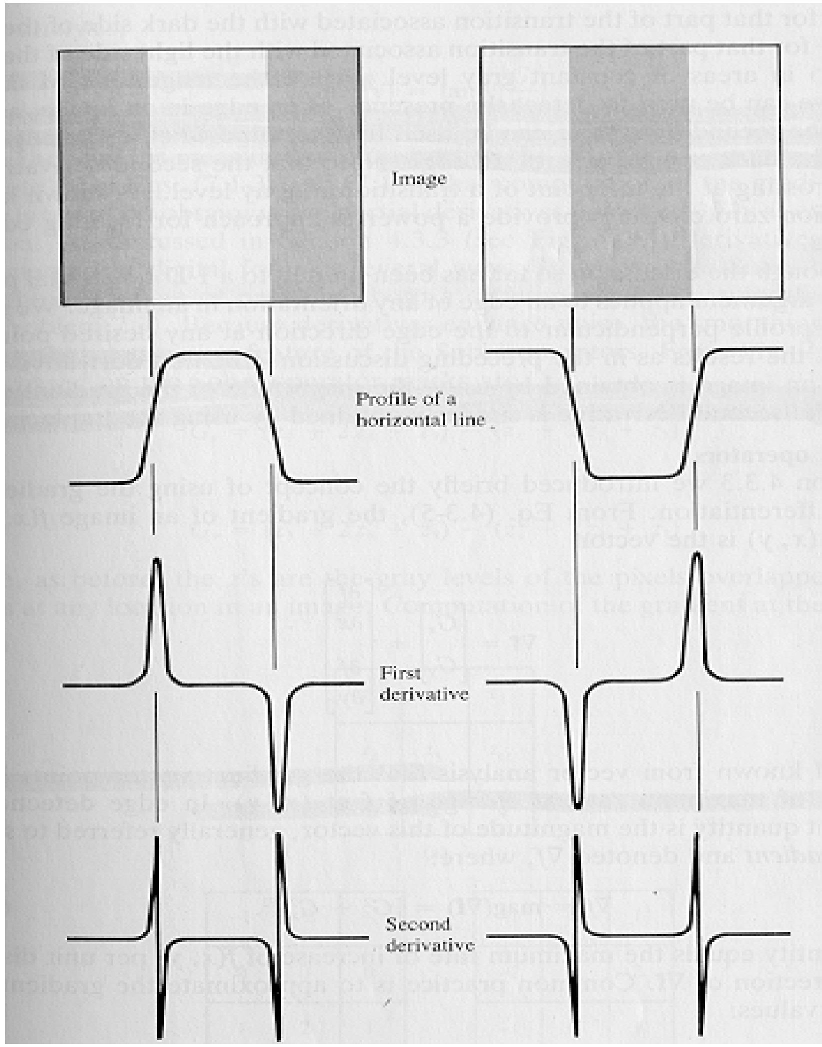
FIGURE 3.36 Illustration of the first and second derivatives of a 1-D digital function representing a section of a horizontal intensity profile from an image. In (a) and (c) data points are joined by dashed lines as a visualization aid.

Derivatives and Edges

- Often, points that lie on an edge are detected by:

➢ Detecting the local **maxima** or **minima** of the first derivative

➢ Detecting the **zero-crossings** of the second derivative



Derivatives of 2-D Digital Functions

- A basic definition of the 1st-order derivatives of the function $f(x,y)$

$$\frac{\partial f}{\partial x}=f(x+1,y)-f(x,y);\frac{\partial f}{\partial y}=f(x,y+1)-f(x,y)$$

- A basic definition of the 2nd-order derivatives of the function $f(x,y)$

$$\frac{\partial^2 f}{\partial x^2}=f(x+1,y)+f(x-1,y)-2f(x,y);$$

$$\frac{\partial^2 f}{\partial y^2}=f(x,y+1)+f(x,y-1)-2f(x,y)$$

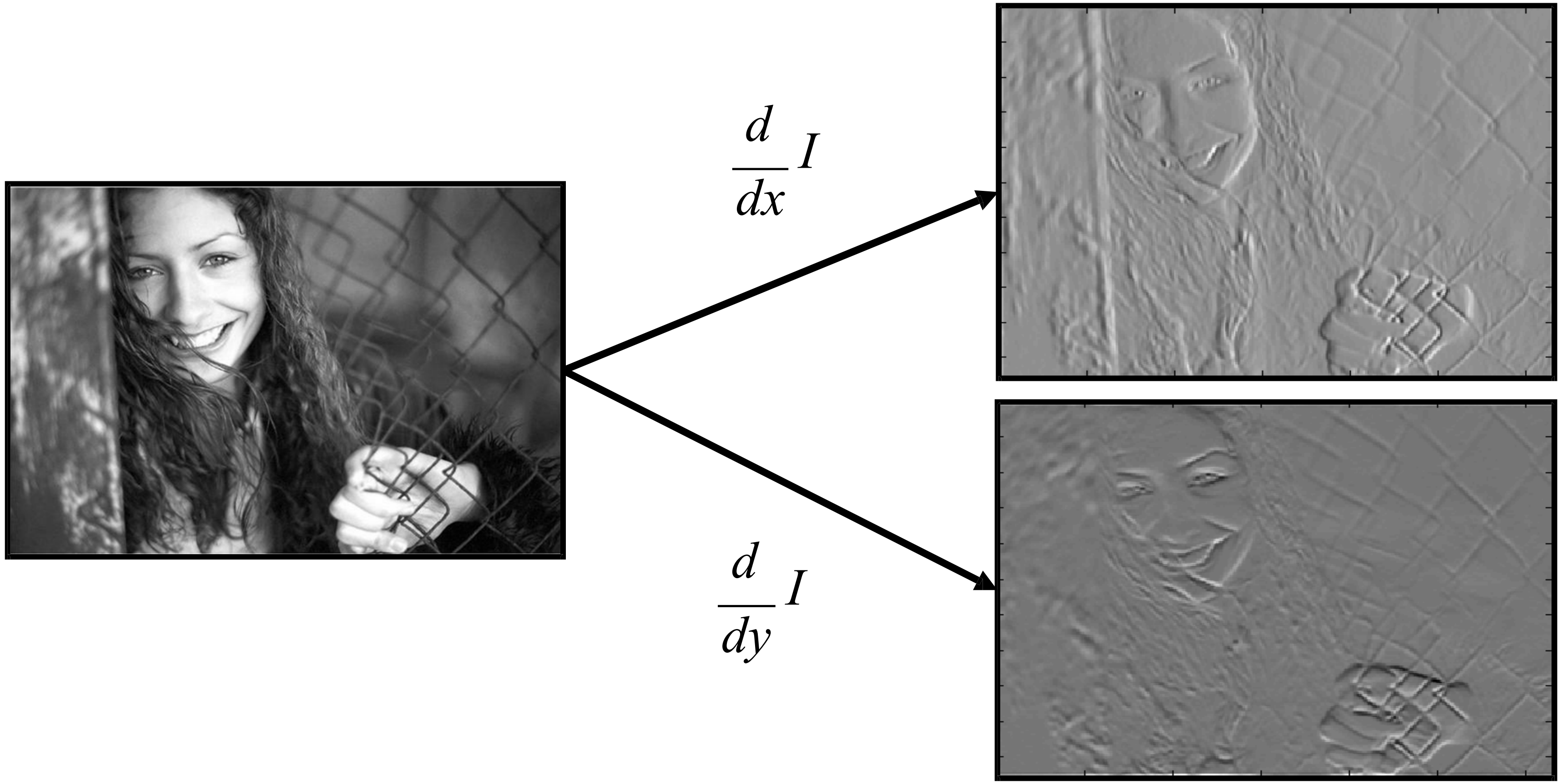
Derivatives of 2-D Digital Functions

- A basic definition of the 1st-order derivatives of the function $f(x, y)$

$$\frac{\partial f}{\partial x} = f(x+1, y) - f(x, y)$$ → sensitive to horizontal edges

$$\frac{\partial f}{\partial y} = f(x, y+1) - f(x, y)$$ → sensitive to vertical edges

Horizontal and Vertical Edges using 1st Derivative



Edge Detection Using 1st order Derivative

$$\frac{\partial f}{\partial x} = f(x+1) - f(x)$$

(not centered at x) mask: $[-1 \quad 1]$

mask M $= [-1, 0, 1]$ (centered at x)

A centered mask detects an edge in both pixels between which an intensity jumps

(upward) step edge

S_1			12	12	12	12	12	24	24	24	24	24
S_1	$\otimes$	M	0	0	0	0	12	12	0	0	0	0

(-1)*12+0*12+1*24=12

(-1)*12+0*24+1*24=12

(downward) step edge

S_2			24	24	24	24	24	12	12	12	12	12
S_2	$\otimes$	M	0	0	0	0	-12	-12	0	0	0	0

(-1)*24+0*24+1*12= -12

(-1)*24+0*12+1*12= -12

The Laplacian and Second Order Derivative Edge Detection

- The Laplacian (as it is known from calculus and partial differential equations) is

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

❖ Sum of the 2nd order derivatives

The Laplacian and Second Order Derivative Edge Detection

- Hence, Laplacian of the image intensity function ("Laplacian 1") $f(x)$ is a sum of its vertical and horizontal 2nd order derivatives

$$\frac{\partial^2 f}{\partial x^2} = f(x+1,y) + f(x-1,y) - 2f(x,y);$$

$$\frac{\partial^2 f}{\partial y^2} = f(x,y+1) + f(x,y-1) - 2f(x,y)$$

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

Edge Detection Using 2nd order Derivative

(upward) step edge

S_1			12	12	12	12	12	24	24	24	24	24
S_1	$\otimes$	M	0	0	0	0	-12	12	0	0	0	0

Zero-crossing

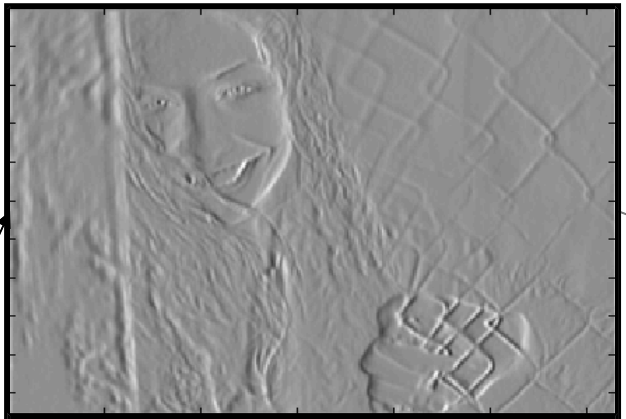
(downward) step edge

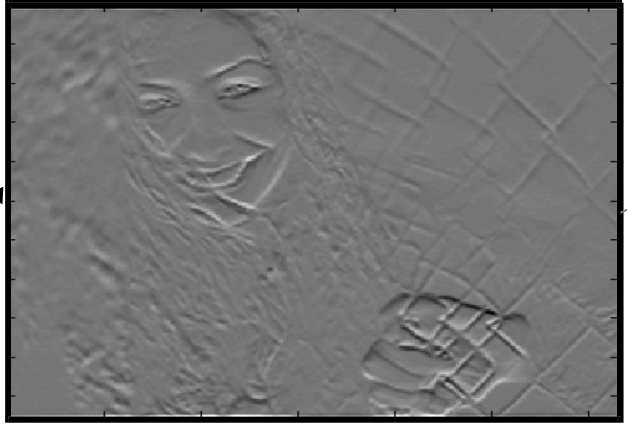
S_2			24	24	24	24	24	12	12	12	12	12
S_2	$\otimes$	M	0	0	0	0	12	-12	0	0	0	0

Zero-crossing

Horizontal and Vertical Edges using Laplacian1

Horizontal edges





Vertical edges

$$\Delta = \sqrt{\left(\frac{d}{dx}I\right)^2 + \left(\frac{d}{dy}I\right)^2}$$

Laplacian 1

The Laplacian

- A global edge detector must be spatially isotropic, it should not depend on any rotation of an image and particular gradient direction
- To detect also diagonal edges, we should involve in the Laplacian the diagonal second-order derivatives (Laplacian 2) $\frac{\partial f}{\partial x \partial y}, \frac{\partial f}{\partial y \partial x}$

$$\nabla^2 f(x,y) = f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) +$$
$$+ f(x-1,y-1) + f(x+1,y-1) + f(x-1,y+1) + f(x+1,y+1) -$$
$$-8 f(x,y)$$

Laplacian Edge Detector

Downward

0	1	0
1	−4	1
0	1	0

1	1	1
1	−8	1
1	1	1

Upward

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

a b
c d

FIGURE 3.37 (a) Filter mask used to implement Eq. (3.6-6). (b) Mask used to implement an extension of this equation that includes the diagonal terms. (c) and (d) Two other implementations of the Laplacian found frequently in practice.

- Laplacian 1

Laplacian 2

Laplacian Detector of Lines Oriented in Various Directions

Horizontal

−1	−1	−1
2	2	2
−1	−1	−1

+45°

2	−1	−1
−1	2	−1
−1	−1	2

Vertical

−1	2	−1
−1	2	−1
−1	2	−1

−45°

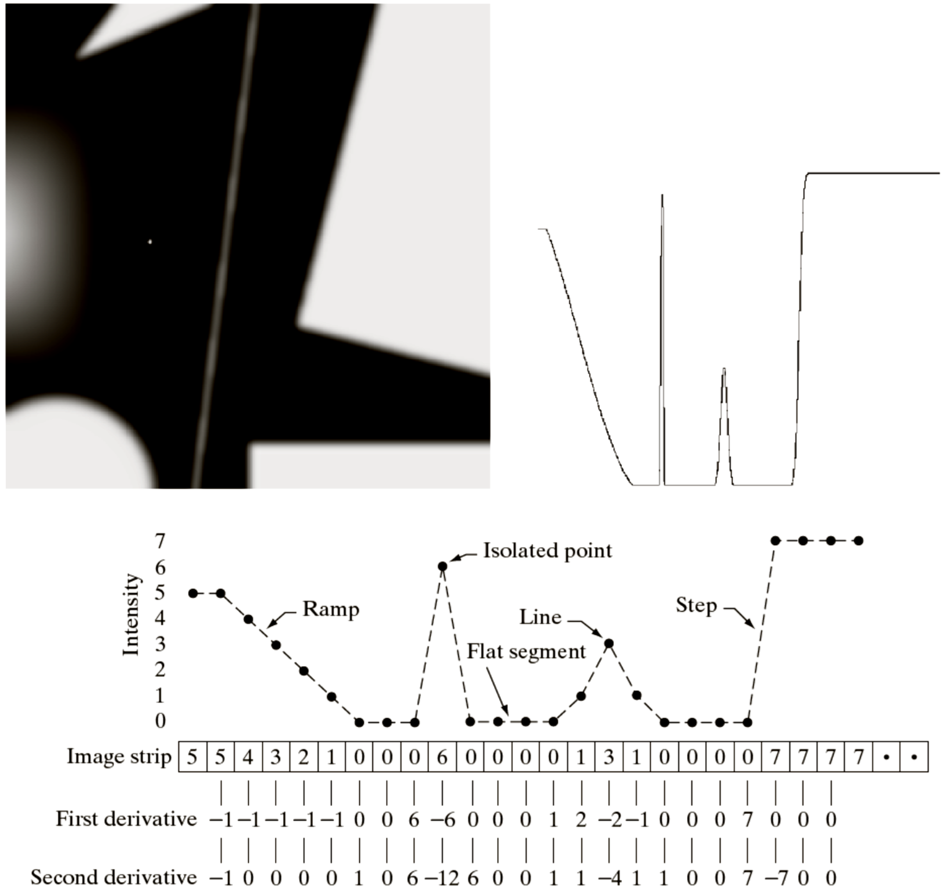
−1	−1	2
−1	2	−1
2	−1	−1

FIGURE 10.6 Line detection masks. Angles are with respect to the axis system in Fig. 2.18(b).

First- and Second- Order Derivatives and Edge Detection

- Laplace edge detectors are based on second-order derivatives
- There are also edge detectors based on first-order derivatives
- Second-order derivatives better detect edges corresponding to the smallest details, they also detect and enhance isolated pixels
- First-order derivatives detect and possibly enhance the intensity jumps “as they are”

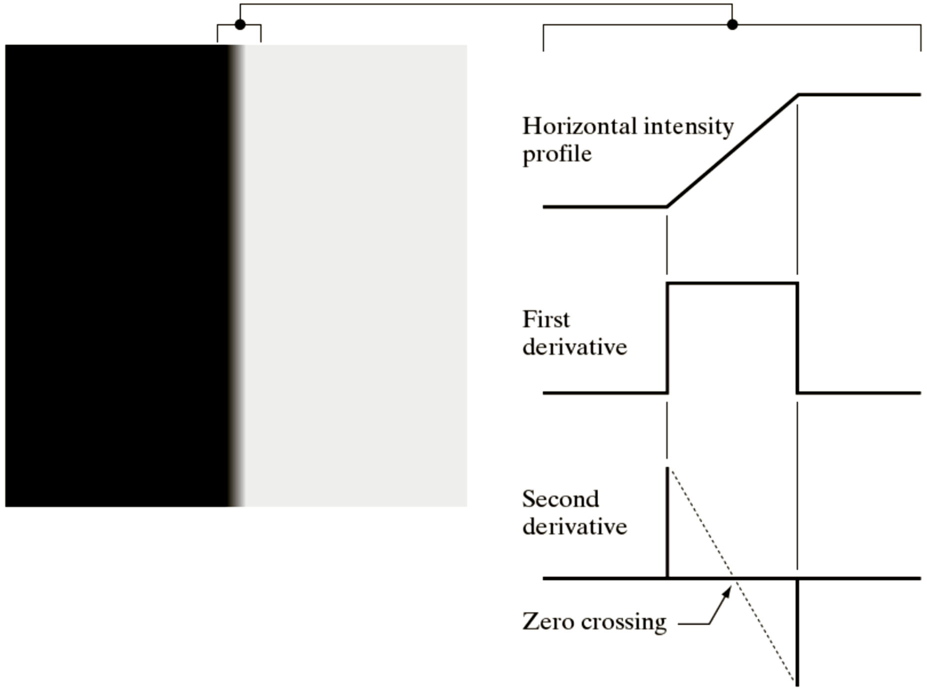
First- and Second- Order Derivatives and Edge Detection



a b
c

FIGURE 10.2 (a) Image. (b) Horizontal intensity profile through the center of the image, including the isolated noise point. (c) Simplified profile (the points are joined by dashes for clarity). The image strip corresponds to the intensity profile, and the numbers in the boxes are the intensity values of the dots shown in the profile. The derivatives were obtained using Eqs. (10.2-1) and (10.2-2).

Distinction between First- and Second- Order Derivatives in Edge Detection



a b

FIGURE 10.10 (a) Two regions of constant intensity separated by an ideal vertical ramp edge. (b) Detail near the edge, showing a horizontal intensity profile, together with its first and second derivatives.

First-Order Derivatives in Edge Detection

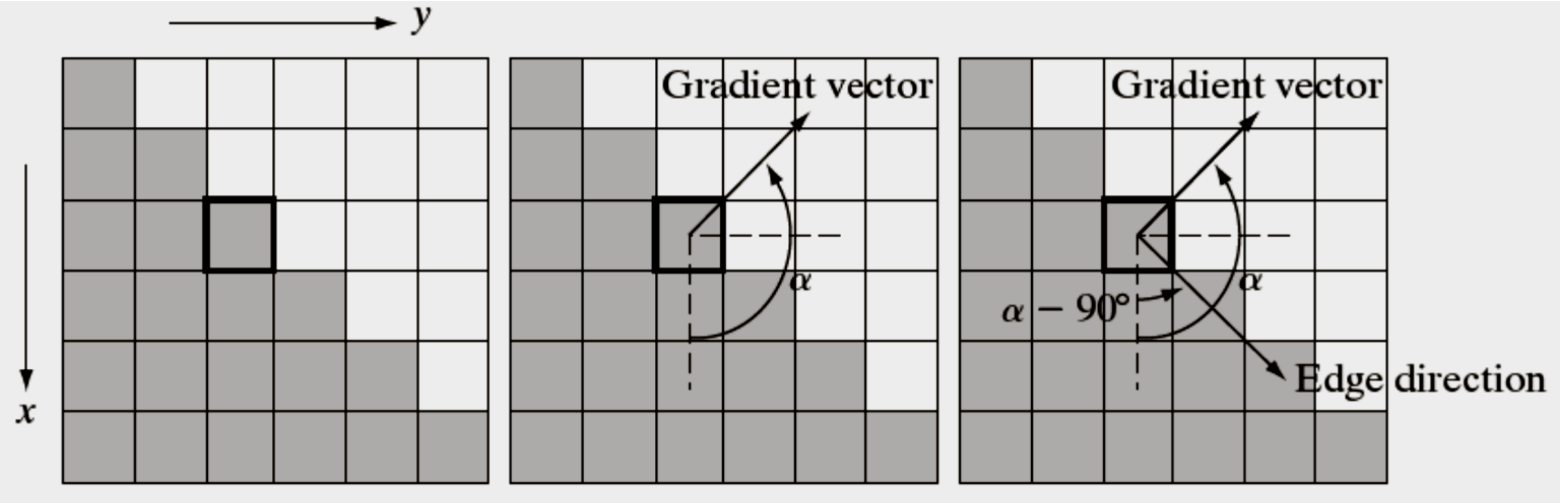
- First-order derivatives in image processing can be implemented using the magnitude of the gradient

$$\nabla f = grad(f) = \begin{bmatrix} g_x \\ g_y \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

- The magnitude (length) of vector ∇f is

$$M(x, y) = mag(\nabla f) = \sqrt{g_x^2 + g_y^2} \approx |g_x| + |g_y|$$

Gradient and Edge



a b c

FIGURE 10.12 Using the gradient to determine edge strength and direction at a point. Note that the edge is perpendicular to the direction of the gradient vector at the point where the gradient is computed. Each square in the figure represents one pixel.

First-Order Derivatives by Roberts (1965)

$$\begin{pmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{pmatrix}$$

$$g_x = (z_6 - z_5); g_y = (z_8 - z_5)$$

$$M(x, y) \approx |z_6 - z_5| + |z_8 - z_5|$$

Robert's derivatives and gradients

$$g_x = (z_9 - z_5); g_y = (z_8 - z_6)$$

$$M(x, y) \approx |z_9 - z_5| + |z_8 - z_6|$$

Disadvantage: no center of symmetry

First-Order Derivatives by Prewitt (1970)

$$\begin{pmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{pmatrix}$$

$$g_x = \frac{\partial f}{\partial x} = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$$

$$g_y = \frac{\partial f}{\partial y} = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$$

$$M(x, y) \approx \left|(z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)\right| +$$

$$+\left|(z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)\right|$$

First-Order Derivatives by Sobel (1970)

$$\begin{pmatrix} z_1 & z_2 & z_3 \\ z_4 & z_5 & z_6 \\ z_7 & z_8 & z_9 \end{pmatrix}$$

$$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$$

$$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$$

$$M(x, y) \approx \left|(z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)\right| +$$

$$+ \left|(z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)\right|$$

Roberts, Prewitt, and Sobel masks

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

Diagonal

−1	0
0	1

Diagonal

0	−1
1	0

Roberts

Horizontal

−1	−1	−1
0	0	0
1	1	1

Vertical

−1	0	1
−1	0	1
−1	0	1

Prewitt

Horizontal

−1	−2	−1
0	0	0
1	2	1

Vertical

−1	0	1
−2	0	2
−1	0	1

Sobel

a
b c
d e
f g

FIGURE 10.14 A 3×3 region of an image (the z's are intensity values) and various masks used to compute the gradient at the point labeled z_5.

Some Disadvantages of Classical Edge Detectors

- Sobel and Prewitt detectors are resulted in “bold” (“thick”) edges where the same intensity jump is detected twice because they are based on the centered masks
- All of them (except Prewitt) give a preference to the pixel of interest (the symmetry center). This may help to suppress noise, to avoid its detection, but simultaneously some edges can be emphasized, while some other can be suppressed

Nonlinear Edge Detectors

- Nonlinear edge detectors are based either on nonlinear operation applied to digital derivatives or nonlinear transformation of an image followed by differentiation

Threshold Boolean Filtering

- Let X be a gray scale image with M+1 intensity levels. It can be broken up using threshold Boolean decomposition is resulted in M binary planes

$$x^{(k)}(i,j) = \begin{cases} 1 & , \text{ if } x(i,j) \ge k \\ 0 & , \text{ otherwise} \end{cases} ; k = 1, \ldots, M$$

- Let $X_{ij}^{(k)}$ be the kth binary plane
- Then filtering operation is determined by $y(i,j) = \sum_{k=1}^{M} F\left(X_{ij}^{(k)}\right)$

where F is the Boolean processing function applied to a local window around a pixel of interest

Edge Detecting Boolean Functions

- The Boolean function for detection of upward intensity jumps

$$F_u = f\begin{pmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{pmatrix} = x_5 \,\&\, \left(\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee \bar{x}_6 \vee \bar{x}_7 \vee \bar{x}_8 \vee \bar{x}_9\right)$$

- The Boolean function for detection of downward intensity jumps

$$F_d = f\begin{pmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{pmatrix} = \bar{x}_5 \,\&\, \left(x_1 \vee x_2 \vee x_3 \vee x_4 \vee x_6 \vee x_7 \vee x_8 \vee x_9\right)$$

Edge Detecting Boolean Functions

$$\begin{pmatrix} 0 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}$$

Upward Edge passes through the center of the window, there is no downward edge

$$\begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Downward Edge passes through the center of the window, there is no upward edge

TBF (Stack) Edge Detection Filter

- Detection of upward brightness jumps is equivalent to

$$g(x,y)=\begin{cases} m=\max\left(f(x,y)-f(x\pm 1,y\pm 1)\right), & \text{if } m>0 \\ 0, & \text{otherwise} \end{cases}$$

- Detection of downward brightness jumps is equivalent to

$$g(x,y)=\begin{cases} m=\max\left(f(x\pm 1,y\pm 1)-f(x,y)\right), & \text{if } m>0 \\ 0, & \text{otherwise} \end{cases}$$

TBF Edge Detection

$$\begin{pmatrix} 35 & 210 & 100 \\ 112 & 120 & 80 \\ 100 & 221 & 220 \end{pmatrix}$$

Upward Edge pass through the center of the window, its intensity is 120-35=85

$$\begin{pmatrix} 35 & 210 & 100 \\ 112 & 120 & 80 \\ 100 & 221 & 220 \end{pmatrix}$$

Downward Edge pass through the center of the window, its intensity is 221-120=101

Edge Detection using Stack Filter

- Simultaneous detection of downward and upward intensity jumps

$$g(x,y)=\max\left|f(x,y)-f(x\pm1,y\pm1)\right|$$

- This is resulted in “bold” (“thick”) edges

Image Break-up Into Binary Planes

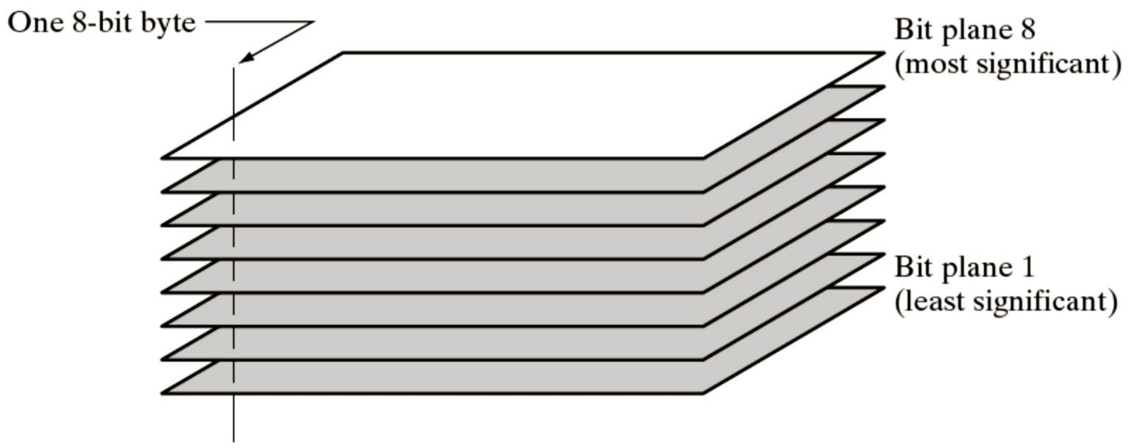


FIGURE 3.13 Bit-plane representation of an 8-bit image.

- Each intensity value in the range 0…255 is presented by one byte, which consists of 8 bits
- Breaking up these bytes into bits, we break up the entire image into 8 binary planes

Precise Edge Detection

- Algorithm:

  1) Breaking up the image into binary planes;

  2) Detection of edges using edge detecting Boolean function;

  3) Assembling the resulting image from the binary planes

Precise Edge Detection

- Upward Edge Detecting Boolean function

$$f\begin{pmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{pmatrix} = x_5 \wedge (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3 \vee \overline{x}_4 \vee \overline{x}_6 \vee \overline{x}_7 \vee \overline{x}_8 \vee \overline{x}_9)$$

- Downward Edge Detecting Boolean function

$$f\begin{pmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{pmatrix} = \overline{x}_5 \wedge (x_1 \vee x_2 \vee x_3 \vee x_4 \vee x_6 \vee x_7 \vee x_8 \vee x_9)$$

- "Global" Edge Detecting Boolean function

$$f\begin{pmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{pmatrix} = \big(x_5 \,\&\, (\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3 \vee \overline{x}_4 \vee \overline{x}_6 \vee \overline{x}_7 \vee \overline{x}_8 \vee \overline{x}_9)\big) \vee \\ \vee \overline{x}_5 \,\&\, (x_1 \vee x_2 \vee x_3 \vee x_4 \vee x_6 \vee x_7 \vee x_8 \vee x_9)$$

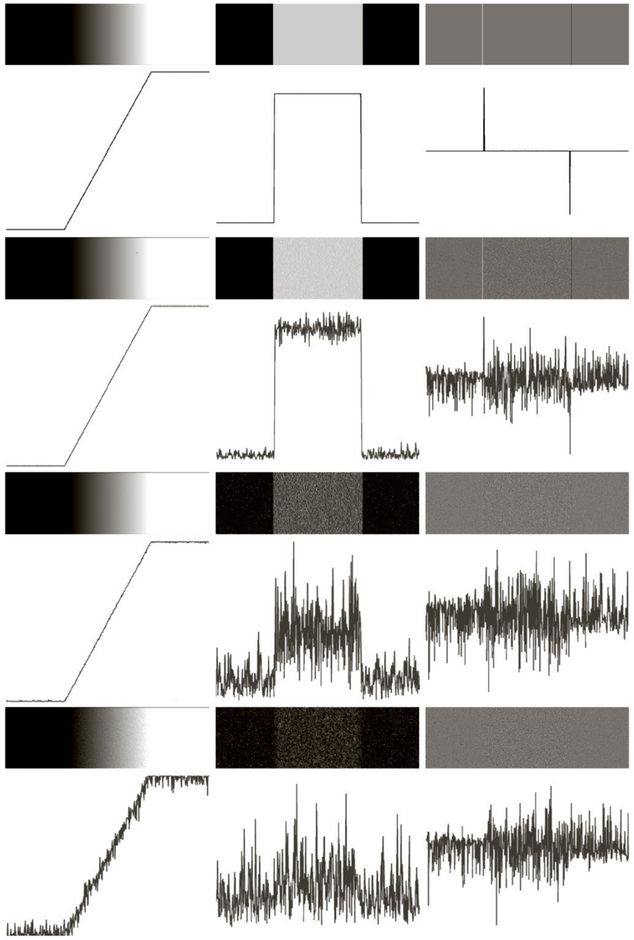
Edged Segmentation using Precise Edge Detection

- Edged Segmentation is edge detection with simultaneous emphasizing of areas with different textures bounded by edges
- To make segmentation using precise edge detection, it is necessary to detect edges in some binary planes preserving some other binary planes unchanged. For example, a good segmentational effect can be achieved by detecting edges in binary planes 0,1,2,3,4,7 and preserving binary planes 5,6 unchanged

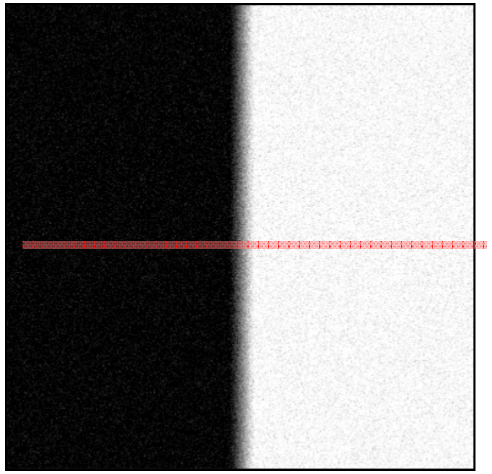
Noise and Edge Detection

- Noise prevents detecting clear edges because edge detectors don’t care of what is behind particular intensity jump – noise or a useful detail

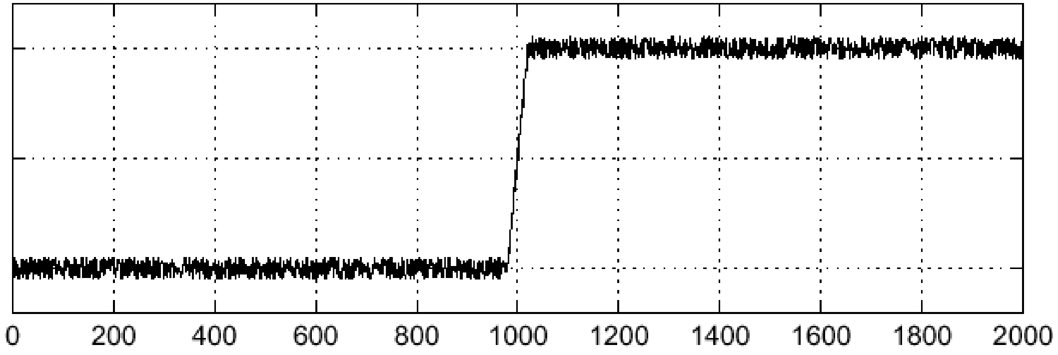
FIGURE 10.11 First column: Images and intensity profiles of a ramp edge corrupted by random Gaussian noise of zero mean and standard deviations of 0.0, 0.1, 1.0, and 10.0 intensity levels, respectively. Second column: First-derivative images and intensity profiles. Third column: Second-derivative images and intensity profiles.



Noise and Edge Detection



$$f(x)$$



$$\frac{d}{dx}f(x)$$